

1                   **CACHING MEMORY CONTENTS INTO CACHE PARTITIONS**  
2                   **BASED ON MEMORY LOCATIONS**

3                   **BACKGROUND OF THE INVENTION**

4                   **Technical Field**

5                   This invention relates generally to computer systems having data and/or  
instructions stored in a memory, and more particularly to such systems in which the data  
and/or instructions can also be temporarily stored from the memory into a cache.

6                   **Description of the Prior Art**

7                   Most modern computer systems include a processor and memory, among other  
components. Data and instructions required for processing by the processor are retrieved  
from the memory. The processor can then store the results of its processing back into the  
memory. Among different types of processors in a given computer system, there usually  
is a central processing unit (CPU), which is the main processor for the system. Multi-  
processor systems contain multiple processors, with no one processor designated as the  
CPU.

8                   Memory access by a processor, however, can be slow. Generally, there is a  
latency associated with each kind of memory, which refers to the length of time from  
when a processor first requests data or an instruction stored in the memory, to when the  
processor actually receives the data or the instruction from the memory. Different  
memory within a computer system may have different latencies. Usually the processor  
itself can process instructions and perform actions faster than the memory can provide  
data and instructions to the processor. This can cause processors to stall when waiting for  
memory responses, resulting in less than ideal performance.

9

To alleviate this problem, many computer systems include one or more caches. A memory cache, or processor cache, is a memory bank that bridges the main memory and a processor. It is faster than the main memory and allows instructions to be fetched and data to be read at higher speeds. The cache essentially contains a subset of the data and instructions that can be found in memory. Instructions and data may be transferred to the cache in blocks, using a look-ahead algorithm. The more sequential the instructions in the routine being accessed, and the more sequential the order of the data being read, the greater the chance the next desired item will still be in the cache, and the greater the improvement in performance. Data reuse also contributes to cache effectiveness. The more often data is reused, the higher the probability it will be in the cache. If data is used infrequently or there is a lot of time between consecutive reuses, there is a low probability it will remain in the cache.

Two common types of caches are known as level 1 (L1) cache and level 2 (L2) cache. An L1 cache is a memory bank built into the processor itself. An L2 cache is a secondary staging area that feeds the L1 cache. Increasing the size of the L2 cache may speed up some applications but may have no effect on others. An L2 cache may be built into the same chip as the processor, reside on a separate chip in a multi-chip package module, or be a separate bank of chips. Caches are typically static random-access memory (SRAM), whereas main memory is generally some variety of slower, more dense dynamic random-access memory (DRAM). Caches may also be provided at higher levels, such as level 3 (L3), and so on, and may also be shared by multiple processors in a multi-processor system.

11           Standard cache allocation policies replace the contents of the cache usually without regard to memory utilization or latency. For example, a least recently used (LRU) policy may replace the data or instruction that was least recently used with a new data or instruction that has been retrieved from memory. Such policies do not concern themselves with how often the newly stored data or instruction may actually be accessed, nor with the latency of retrieving this data or instruction from the memory itself. This can lead to a slowdown in system performance, due to ineffective use of the cache.

12           For example, in streaming data applications, such as streaming video or streaming audio applications, the data retrieved from the memory is usually used only once, and then not used again. A typical cache will dutifully cache this data, however, as it is retrieved by the processor. This negates the usefulness of the cache, because the cache is desirably meant for data that will be used often by the processor. Online transaction processing also tends to have large sets of data that are used infrequently and a small set of data that is used frequently.

13           As another example, some memory of the system may be high-speed memory, with relatively low latencies. Caching the contents of such memory, as compared to caching the contents of higher-latency memory, may cause an overall decrease in system performance. This is because the performance benefits of caching lower-latency memory are less than those of caching higher-latency memory. A typical cache, however, does not discern the latency of memory when caching data or instructions from the memory.

14           For these described reasons, as well as other reasons, there is a need for the present invention.

15                                   **SUMMARY OF THE INVENTION**

16                   The invention relates to caching memory contents into cache partitions based on their locations. In a method of the invention, a location of a line of memory to be cached in a cache is determined. The cache is partitioned into a number of cache sections. The section for the line of memory is determined based on the location of the line of memory as applied against a memory line location-dependent allocation policy. The line of memory is then stored in the section of the cache determined.

17                   In a system of the invention, there is a cache, a number of cache sections, an allocation policy, and a mechanism. The cache is for caching lines of memory, including a line to be cached. The cache is partitioned into the number of cache sections. The allocation policy specifies which of the lines of memory are allocated to which of the cache sections, based on locations of the lines of memory. The mechanism determines the section for the line of memory to be cached based on a location of the line as applied against the allocation policy. The mechanism then stores the line of memory in the section determined.

18                   An article of manufacture of the invention includes a computer-readable medium and means in the medium. The means in the medium is for determining one of a number of cache sections into which a cache has been partitioned to store a line of memory to be cached. The means makes this determination based on a location of the line of memory as applied against a memory line location-dependent cache allocation policy. The means is also for storing the line of memory in the section determined. Other features and advantages of the invention will become apparent from the following detailed description

of the presently preferred embodiment of the invention, taken in conjunction with the accompanying drawings.

## BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a flowchart of a method according to a preferred embodiment of the invention, and is suggested for printing on the first page of the issued patent.

FIG. 2 is a diagram of a computer system in conjunction with which embodiments of the invention may be implemented.

FIG. 3 is a diagram of a memory having a number of lines, each having a unique location in the memory, that are allocated to different cache sections into which a cache has been partitioned, as specified by a cache allocation policy, according to an embodiment of the invention.

FIG. 4 is a diagram of a system for storing lines of memory in different cache sections into which a cache has been partitioned, as specified by a cache allocation policy, according to an embodiment of the invention.

FIG. 5 is a flowchart of a method according to an embodiment of the invention that is more detailed than the method of FIG. 1, but is still consistent with the method of FIG. 1.

## DESCRIPTION OF THE PREFERRED EMBODIMENT

### Overview

FIG. 1 shows a method 100 according to a preferred embodiment of the invention. The method 100 indicates the manner by which the preferred embodiment can cache memory contents in different cache sections into which a cache has been partitioned,

based on the locations of the memory contents as applied against a cache allocation policy. The method 100 can be implemented as a means in a computer-readable medium stored on an article of manufacture. The medium may be a recordable data storage medium, or another type of medium. Other components described in the detailed description can also be implemented as such means.

28

First, a location of a line of memory to be cached in one of a number of sections into which a cache has been partitioned is determined (102). A line of memory generally refers to a part of the memory, where the memory can be divided into separate lines. A line may be a byte of memory, a word of memory, and so on, in different embodiments of the invention. Each line of memory is preferably addressable by a unique address that specifies a particular location in the memory. That is, the location of the line of memory is preferably, but not necessarily, the unique address of the line of memory.

29

The cache is partitioned into a number of cache sections. The cache sections may be the same size, or more likely, may be differently sized. A memory line location-dependent cache allocation policy determines which lines of memory are to be cached in which of these cache sections, based on the locations of the lines of memory. For example, streaming multimedia data that is stored in certain locations of the memory may be allocated for caching in a relatively small section of the cache, whereas other, highly utilized data that is stored in other locations of the memory may be allocated for caching in a relatively large section of the cache. The cache allocation policy, as well as the size of each cache partition, may each be statically set, or may dynamically change over time.

30

The method 100 determines the cache section for the line of memory to be cached, based on its location as applied against the cache allocation policy (104). That is,

the location of the line of memory dictates in which section of the cache it is to be stored. The line of memory is then stored in this cache section (106). The method 100 thus achieves caching different memory lines into different cache sections based on the locations of the memory lines. Memory lines having low latency and/or infrequent usage may be allocated to a relatively small cache section, based on their locations in memory, so that not many such lines are cached in the cache. Conversely, memory lines having high latency and/or frequent usage may be allocated to a relatively large cache section, based on their locations in memory, so that many such lines are cached.

### Technical Background

FIG. 2 shows one example of a computer system 200 in conjunction with which embodiments of the invention may be practiced. Other computer systems may also be used in conjunction with embodiments of the invention. Only those components of the computer system 200 relevant to embodiments of the invention are illustrated, and other components may be included in the system 200 that are not illustrated or described herein. The system 200 specifically includes a central processing unit (CPU) 202, system (main) memory 204, a level 1 (L1) cache 206, and a level 2 (L2) cache 208. Other computer systems that may be used in conjunction with embodiments of the invention include multi-processor systems that do specifically include a processor designated as the CPU, and/or systems that also have more or less levels of cache, such as a level 3 (L3) cache, and so on.

The CPU 202 is one type of processor, and the system 200 may include other types of processors as well, such as co-processors, mathematical processors, service processors, and so on. The system, or main, memory 204 is the primary memory in

which data and computer instructions are stored for access by the CPU 202. The memory 204 may have different latencies, and desirably is significantly larger than the size of either the L1 cache 206 or the L2 cache 208. The term memory is used generally herein, and is meant to encompass any type of storage, such as hard disk drives, and so on, although the detailed description itself is particular to what is generally known as semiconductor or solid-state memories.

The L1 cache 206 as shown in FIG. 2 is within the CPU 202, and is smaller in size, but faster, than the L2 cache 208. The L1 cache 206 may alternatively be on a separate chip than the CPU 202, and so on. The L2 cache may or may not be located on the same chip as the processor. As indicated, it is larger in size, but slower, than the L1 cache 206. Embodiments of the invention can be implemented in conjunction with the L1 cache 206, the L2 cache 208, both the caches 206 and 208, and/or other types of caches as well, including shared caches in multiprocessor systems.

### **Cache Partitioning and System**

FIG. 3 shows an example of partitioning a cache into different sections, and how different locations of a memory are allocated to different cache sections based on a cache allocation policy. In the system 300, a memory 302 is divided for sake of description into sections 302a and 302b, where memory section 302a covers lines having memory locations described as locations 0-m, and memory section 302b covers lines having memory locations described as locations n-z. For example, where locations of the memory 302 are specified by unique addresses, the memory section 302a may cover memory locations hexadecimal 0x0000 through hexadecimal 0xC000, and the memory section 302b may cover memory locations hexadecimal 0xC001 through 0xFFFF. The

difference between these memory locations may be that those covered by the memory section 302a are high-latency memory, whereas those covered by the memory section 302b are low-latency memory, for instance.

37

The cache 304 is partitioned into two sections, the cache section 304a and the cache section 304b. The cache section 304a is larger than the cache section 304b, and thus the cache section 304a can cache a larger number of memory lines than the cache section 304b can. The memory line location-dependent cache allocation policy 306 specifies which lines of the memory 302 are cached in which of the sections 304a and 304b of the cache 304. As shown in FIG. 3, for example, the memory locations covered by the memory section 302a are allocated to the larger cache section 304a, whereas the memory locations covered by the memory section 302b are allocated to the smaller cache section 304b. This may be because the high-latency memory of the memory section 302a should be cached more often than the low-latency memory of the memory section 302b, for performance reasons.

38

As another example, the memory locations covered by the memory section 302a may have high temporal locality, and the memory locations covered by the memory section 302b may have low temporal locality. Therefore, the memory locations covered by the memory section 302a are allocated to the larger cache section 304a, whereas the memory locations covered by the memory section 302b are allocated to the smaller cache section 304b, for performance reasons. Furthermore, both the allocation policy 306 in terms of which memory locations are to be cached by which cache sections 304a and 304b, and the sizes of the cache sections 304a and 304b, can be static or dynamic.

39

FIG. 4 shows a system 400 according to an embodiment of the invention. The system 400 includes a mechanism 402 that has a memory line location-dependent cache policy 404, a memory 406, and a cache 408. The memory 406 has a number of different lines, each of which has a unique location, such as a unique address. The cache 408 is partitioned into a number of cache sections 408a, 408b, . . . , 408n, which may be differently sized. The allocation policy 404 specifies which of the lines of the memory 406 are allocated to which of the cache sections 408a, 408b, . . . , 408n, based on the locations of the lines of the memory 406.

40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100  
101  
102  
103  
104  
105  
106  
107  
108  
109  
110  
111  
112  
113  
114  
115  
116  
117  
118  
119  
120  
121  
122  
123  
124  
125  
126  
127  
128  
129  
130  
131  
132  
133  
134  
135  
136  
137  
138  
139  
140  
141  
142  
143  
144  
145  
146  
147  
148  
149  
150  
151  
152  
153  
154  
155  
156  
157  
158  
159  
160  
161  
162  
163  
164  
165  
166  
167  
168  
169  
170  
171  
172  
173  
174  
175  
176  
177  
178  
179  
180  
181  
182  
183  
184  
185  
186  
187  
188  
189  
190  
191  
192  
193  
194  
195  
196  
197  
198  
199  
200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211  
212  
213  
214  
215  
216  
217  
218  
219  
220  
221  
222  
223  
224  
225  
226  
227  
228  
229  
230  
231  
232  
233  
234  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244  
245  
246  
247  
248  
249  
250  
251  
252  
253  
254  
255  
256  
257  
258  
259  
260  
261  
262  
263  
264  
265  
266  
267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323  
324  
325  
326  
327  
328  
329  
330  
331  
332  
333  
334  
335  
336  
337  
338  
339  
340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447  
448  
449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537  
538  
539  
540  
541  
542  
543  
544  
545  
546  
547  
548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
600  
601  
602  
603  
604  
605  
606  
607  
608  
609  
610  
611  
612  
613  
614  
615  
616  
617  
618  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
740  
741  
742  
743  
744  
745  
746  
747  
748  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
840  
841  
842  
843  
844  
845  
846  
847  
848  
849  
850  
851  
852  
853  
854  
855  
856  
857  
858  
859  
860  
861  
862  
863  
864  
865  
866  
867  
868  
869  
870  
871  
872  
873  
874  
875  
876  
877  
878  
879  
880  
881  
882  
883  
884  
885  
886  
887  
888  
889  
890  
891  
892  
893  
894  
895  
896  
897  
898  
899  
900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
970  
971  
972  
973  
974  
975  
976  
977  
978  
979  
980  
981  
982  
983  
984  
985  
986  
987  
988  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
999  
1000

For a line of the memory 406 to be cached in the cache 408, the mechanism 404 determines to which of the sections 408a, 408b, . . . , 408n the line has been allocated by applying the line's location against the allocation policy 404. The mechanism 404 then stores this line of the memory 406 in the determined cache section. The mechanism 404 may also periodically adjust the allocation policy 404 and/or the sizes of the cache sections 408a, 408b, . . . , 408n, if the policy 404 and/or the cache section sizes are dynamic and not static. The mechanism 404 can further perform any of the functionality of a method according to an embodiment of the invention, such as the method 100 of FIG. 1 that has been described, and the method 500 of FIG. 5 that will be described.

## Method

41

42

FIG. 5 is a flowchart of a method 500 according to an embodiment of the invention. The method 500 is more detailed than but still consistent with the method 100 of FIG. 1. As has been indicated, at least some functionality of the method 500 can be implemented as a means in a computer-readable medium of an article of manufacture.

As has also been indicated, at least some functionality of the method 500 can be performed by the mechanism 402 of the system 400 of FIG. 4.

43 First, a memory line location-dependent cache allocation policy is set (502), and the size of each of a number of cache sections into which a cache has been partitioned is set (504). A line of memory to be cached is then received (506). This may be a line of memory that has been requested and received or otherwise retrieved by a processor, for example. The location of the line of memory is determined (102), by, for instance, examining or otherwise determining the address for the line in memory. The cache section for the line is also determined, based on its location as applied against the cache allocation policy (104). For instance, the cache section to which the location of the line of memory is allocated according to the cache allocation policy is determined.

The line of memory is then stored in this section of the cache (106). This is accomplished first by determining whether the section is full (508). If the cache section is not full, the line of memory is simply added to this section (510). However, if the cache section is full, then a replacement line of memory that is currently stored in this cache section is selected (512). For example, the least recently used (LRU) of the lines of memory stored in the cache section may be selected as the replacement line. Other policies, approaches, and algorithms may also be used to select the replacement line of memory. The replacement line is then replaced by the line of memory to be stored in the section of the cache (514).

45 The method 500 proceeds to dynamically adjust the allocation policy on a periodic basis (516). This is optional, and is performed only where the allocation policy is not static. The allocation policy is adjusted to change which of the lines of memory are

allocated to which of the sections of the cache, based on the locations of the memory lines. Such adjustment may be performed based on the temporal locality of the memory line locations allocated to each cache section, as has been monitored over time; based on the reutilization of the memory line locations allocated to each cache section, as has been monitored over time; or, based on other or additional factors.

46 The method 500 can also dynamically adjust the size of each section into which the cache has been partitioned on a periodic basis (518). This is optional as well, and is performed only where the sizes of the cache sections are not static. The sizes of the cache sections are adjusted to essentially allocate more or less of the cache to particular lines of memory based on the locations of the memory lines. Such adjustment may be performed based on the utilization of each cache section, as has been monitored over time; based on fullness of each cache section, as has been monitored over time; or, based on other or additional factors. The method 500 is then done.

#### 47 **Advantages over the Prior Art**

48 Embodiments of the invention allow for advantages over the prior art.

Embodiments of the invention provide for caching differently located lines of memory in different sections of a cache. This means that, for example, memory from one location may be cached in one section, whereas memory from another location may be cached in another section. Ultimately, such cache partitioning allows for more efficient use of the cache. Low-latency and/or infrequently used memory, or its contents, may be allocated for caching to a much smaller cache section than high-latency and/or frequently used memory, such that the former kind of memory does not flush the latter type of memory from the cache when requested by the processor or otherwise received for caching.

49

**Alternative Embodiments**

50

It will be appreciated that, although specific embodiments of the invention have been described herein for purposes of illustration, various modifications may be made without departing from the spirit and scope of the invention. For example, embodiments of the invention have been substantially described in relation to solid-state or semiconductor memories. However, the term memory has been used generally herein, such that it is inclusive of other types of storage, such as hard disk drives, and so on, as well. Accordingly, the scope of protection of this invention is limited only by the following claims and their equivalents, as their claim limitations in some instances have been specifically defined in the detailed description.